Interactive paper tearing

Camille Schreck, Damien Rohmer, Stefanie Hahmann

EUROGRAPHICS 2017 / L. Barthe and B. Benes
(Guest Editors)

Volume 36 (2017), Number 2

Interactive paper tearing

Camille Schreck[1,3], Damien Rohmer[1,2], Stefanie Hahmann[1]

[1] Univ. Grenoble Alpes & CNRS (LJK), Inria. [2] CPE Lyon, Univ. lyon. [3] IST Austria.

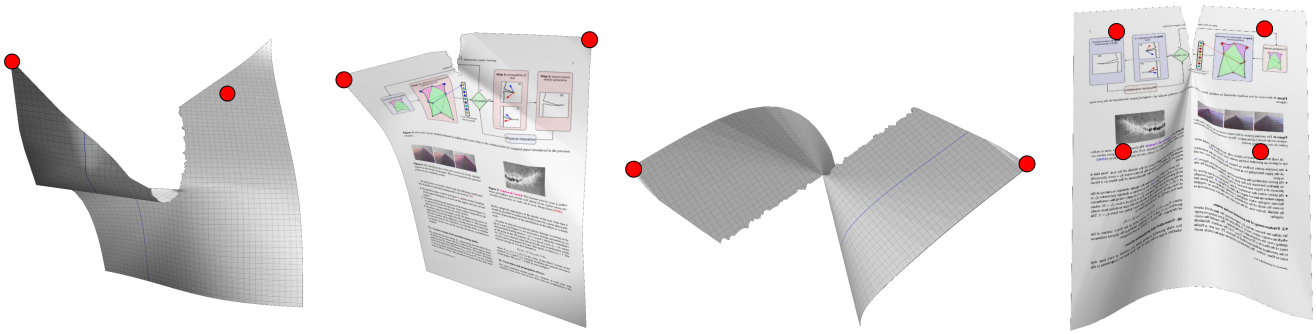


Figure 1: *Interactive tearing of pieces of paper. The path of a tear follows a geometrical curve but also presents stochastic details.*

Abstract

We propose an efficient method to model paper tearing in the context of interactive modeling. The method uses geometrical information to automatically detect potential starting points of tears. We further introduce a new hybrid geometrical and physical-based method to compute the trajectory of tears while procedurally synthesizing high resolution details of the tearing path using a texture based approach. The results obtained are compared with real paper and with previous studies on the expected geometric paths of paper that tears.

Categories and Subject Descriptors (according to ACM CCS): I.3.7 [Computer Graphics]: Three-Dimensional Graphics and Realism—Animation

1. Introduction

Almost everyone has already had occasions to bend, crumple, and tear up a sheet of paper. This very common material, which starting from an initially flat geometry, can undergo a complex geometric deformation when manipulated by hand. Indeed, a sheet of paper exhibits strong mathematical and physical properties, such as a non-smooth developable geometry or its microscopical fiber structure. It may furthermore undergo irreversible damages upon deformation ending up in crumpled and/or torn form. Modeling and animating efficiently such a shape in the virtual world is thus challenging mainly because of the following characteristics of paper: First, a sheet of paper can be represented as a developable surface, which can be flattened onto a plane without distortion, i.e. without stretch nor compression. It is said to be locally isometric to its 2D pattern. This strong geometric property translates into non-linear constraints of length preservation which imply the use of costly algorithms. Second, as mentioned above, at macroscopic scale this surface may not be smooth, exhibiting sharp creases and singularities appearing dynamically when being crumpled. These sharp features are noticeably hard to model using standard triangle meshes as a too coarse sampling or badly oriented connectivity are likely to be seen as visual artifacts. As a consequence, very dense meshes or adaptive remeshing are usually required, increasing even more the computational cost.

Proposing an interactive tool enabling to handle deformation behaviors as different as bending, crumpling and tearing could be highly valuable for both digital artists who need to model and animate the virtual paper, or for digital environment appli-

cations such as virtual and augmented reality, or video games, where the user being immersed into the scene can manipulate the virtual material. While interactive paper bending and crumpling acting under compressive forces have already been studied [BW07, KZC09, HY14, SRH*15], interactive tearing behavior of paper under stretching forces has not yet been addressed.

In this work, we propose a new method to efficiently model paper tearing in the context of an interactive modeling session. The method takes as input a flat sheet of paper. Users can then dynamically distort it and tear it up, the deformation being controlled by the compression or tension forces resulting from the movement of their finger tips applied to the sheet. Note that when you tear the actual paper with your hands, you can naturally control the direction of the tear. One of our objectives is to allow the same kind of intuitive control over our virtual tear. Our approach is associated with bending and crumpling of the paper and thus enables to model common real life scenarios of paper manipulation. While other previous approaches usually require either a fine meshing along the tear in order to reduce dependence from the mesh connectivity [PNdJO14], or require to embed the mesh into a low resolution proxy [KMB*09], our approach, instead, can efficiently compute the tear in an arbitrary direction on the surface mesh. By assuming the specific scenario where a sheet of paper is manipulated by the user's hands, we derive our geometry-based algorithm to compute the propagation of a tear from observations made in material sciences and fracture mechanics. By using a dedicated coarse, yet accurate, representation of the developable surface of paper we finally achieve our goal of realistic looking tears of paper at interactive time rates. Our contributions further include a geometrically-based criteria to select a small subset of candidate vertices where to start a tear, an efficient computation of the tearing direction, and a texture-based appearance of fine details along the tear.

2. Related works

Thin sheet tearing, and more generally, thin sheet fracture have been studied in the Computer Graphics field using two main approaches: the physically-based methods in order to simulate the fracture of different material, and the procedural approaches to generate plausible fracture behaviors and geometries.

Physically-based fracture simulation
The early work of Terzopoulos and Fleischer [TF88] is the first to propose a tearing model for thin sheets computed using a breakable mass-spring model. While the approach is conceptually simple, the fracture orientation heavily depends on the mesh connectivity. O'Brien and Hodgins [OH99] propose a continuous tetrahedron based finite element model handling explicitly this fracture orientation for brittle objects. It has been extended later to ductile material [OBH02]. Solid object fracture has been subject to specific studies such as efficient formulation for FEM meshes [BHTF07], integral formulation [ZBG15], or adapted to point set data [GMD12]. [HJST13] proposes to solve ductile fracture using the so called *Griffith's energy* with a level set method. However, such optimizations are restricted to non deformable models. Interactive time rates have recently been achieved using modal analysis [GMD12]. Gingold *et al.* [GSH*04] propose a fracture formulation dedicated to discrete deformable triangle meshes, which is thus adapted to thin sheets fracture. Pfaff *et al.* [PNdJO14] developed a thin sheet fracture model based on a local stress analysis on top of a local mesh adaptation algorithm [NSO12]. It achieves highly detailed results which can be applied to paper tearing as well, but its computational cost prevents the applications within an interactive scenario.

Procedural fracture model
Some previous works use procedural methods for fracturing volumetric objects. For example, fracture patterns can be computed using a fast procedural method, although the kinematics are still usually computed using a physical model. A popular method for cheap fracture computing, dedicated to explosion modeling, is to use pre-fractured objects that break along a pre-defined pattern [MCK13]. The fracture may also be spread from an initial location using a procedural algorithm [NF99]. This kind of approaches enables fast fracture simulation but cannot be applied to paper as the path of the tear explicitly depends on the motion of the hands tearing the paper. Lejemble *et al.* [LFD*15] proposed an entire procedural method to compute the tearing of paper in a very specific scenario where only four fingers with fixed positions on the sheet or paper are able to tear it.

Fracture details
Different approaches have been proposed to create realistic details at the edges of a tear. Metaaphanon *et al.* [MBCN09] model frayed edges of woven fabrics being torn by representing the two layers of interwoven yarn of the cloth. Although paper may be seen as a structure of interwoven fibers, the small size and stochastic position and orientation of the fibers make this kind of approach hardly efficient for tearing paper. Busaryev *et al.* [BDW13] simulate multi-layered thin material by superposing several thin shells. A mixed approach, proposed by Chen *et al.* [CYFW14], is to refine a 3D fracture animation, computed for example with a coarse physical simulation, by adding procedural details based on a 3D texture representing the strength field of the material (i.e. its ability to resist to fracture). Highly detailed fracture of inhomogeneous material can thus be computed at low cost. Physical accuracy is limited by the coarse initial simulation, and the collision handling of the refined parts may prove to be problematic. Yet this concept seems to be well-adapted to represent small details of the torn fibrous structure. This method inspired the generation of additional 2D details for the procedural method by [LFD*15]. We use a similar approach to generate detailed tearing paths, but improve it by efficiently employing textures.

3. Method overview

The main idea of this work is grounded in several observations from studies made in material sciences and fracture mechanics. First, by reviewing a set of experimental and theoretical studies of tearing paths for different set-ups, such as a cylinder being pushed though a sheet [ARR05] or two points pulling on a tear [OKe94], Roman [Rom13] shows that a tearing path on the 2D pattern of inextensible thin plates is remarkably explained by geometrical criteria. Therefore material properties and history of elastic deformation may have only few impact on the resulting fracture path. Secondly, as explained in [Fre98, LG03], the tear propagation is mainly influ-

enced by the stress field surrounding the tip of the tear. This leads to the general idea of our method based on describing the tear propagation path through local geometrical criteria around the tip of the tear in order to derive an efficient algorithm.

The *Griffith criterion, derived from [Gri21] is used in numerous former and recent works about fracture mechanics, e.g. [SP74, AL85, CFM09] for predicting the propagation of a tear or fracture. Let us suppose a sheet of paper with constant thickness e, this criterion relates the propagation of tears to two quantities: the energy release rate* E_r *(*$J\,m^{-2}$*) expressing the amount of energy released by a infinitesimal fracture propagation and divided by e and the crack resistance force, also called fracture energy* E_f *(*$J\,m^{-2}$*) expressing the magnitude of the force divided by e required to generate a fracture, and can be seen as expressing the intrinsic resistance of the material to tearing.*

Griffith criterion states that a tear is created when

$$E_r \geq E_f \ . \tag{1}$$

Note that the associated computation of these quantities in our scenario will be described in Section 4 and Section 5. Some works propose to compute the direction of propagation as the direction that maximizes the energy released rate E_r [BFM08], i.e. the direction which first satisfies the *Griffith criterion* when forces applied gradually increase.

We derive our geometric propagation criterion from Griffith's work and a specific case studied theoretically and experimentally by O'Keefe [OKe94] and Roman [Rom13], where a sheet of paper is torn apart when pulled by two fingers on both sides of the tear, see Figure 2(left). O'Keefe notes that the propagation of the tear can be fully described by the positions of the two fingers, $\bar{A}$ and $\bar{B}$, and the crack tip $\bar{p}$ *on the 2D pattern* of the sheet of paper. More precisely, in the case of isotropic paper, the direction of propagation that maximizes the energy release rate E_r is given by the *bisector between the two line segments* linking the fingers' positions to the tip $\bar{p}$ of the tear as seen in Figure 2(right). The trajectories are thus hyperbola with $\bar{A}$ and $\bar{B}$ as focal points. The maximized energy release rate E_r can then be formulated as

$$E_r = F \ \ 2\cos\left(\frac{\theta}{2}\right)/e, \tag{2}$$

where F is the amplitude of the force applied on the pulling points, $\theta \in [0, 2\pi]$ is the angle between the two segments linking the finger positions to the tip of the tear, and e is the paper thickness, which is supposed to be a constant defined by the user in our implementation.

Our method aims at applying these ideas to a more general scenario, where tearing may be initiated and propagated using more than only two positional constraints, while still seeking to describe the tearing propagation direction within the 2D pattern domain for efficiency reasons. Our insight is to analyze the distribution of forces acting locally on the tip of the tear to extract two main, approximately equivalent opposite forces. These two opposite forces, seen as generated by two virtual fingers, will be used to define the tearing direction following [OKe94] as the bisector of their directions expressed on the 2D pattern. Note that our assumption of locally equivalent forces is restricted to the cases where multiple positional constraints are pulling apart the sheet of paper which is coherent with our interactive tearing scenario, but may no longer be suitable for other scenarios such as volumetric projectiles passing through the sheet of paper, or when a part of the sheet of paper is stuck on an underlying surface.

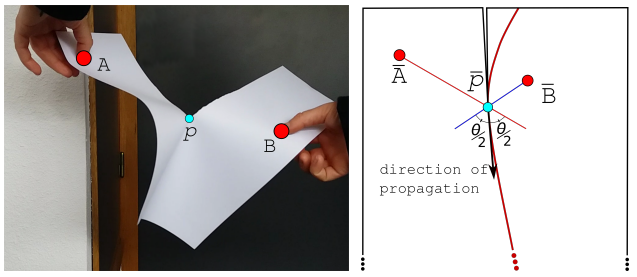


Figure 2: *(left) Two points on a flat sheet are pulled away from each other. The tip of the tear p propagates when the sheet is pulled by points A and B. (right) On the 2D pattern, the direction of propagation is the bisector of the lines* $(\bar{A}, \bar{p})$ *and* $(\bar{B}, \bar{p})$*. The trajectory is then a hyperbola (in red), with focal points in* $\bar{A}$ *and* $\bar{B}$*.*

In order to apply this tearing method for interactive purposes, we need to base our computations on an efficient deformable model of paper material. Our method can be seen as an extension of the interactive model developed by Schreck *et al.* [SRH*15]. This model describes the isometric, and thus developable, non-smooth surface of a deformed sheet of paper as a set of generalized cones and planar surface parts. Each generalized cone is defined as a set of rulings having a single point, the so-called apex, in common. This model can handle both smoothly bent and crumpled papers as response to compressive forces. A smooth shape is defined by piecewise generalized cones with the apex outside the surface, see Figure 3. A crumpled shape includes also conical pieces whose apices are inside the surface. Those apices represent so-called interior *singular points* –the surface is not smooth (C^2) anymore– caused by damage done to the fibrous structure while crumpling. This paper model also includes so-called *junction points*, which designate the junction between curved and planar regions on the boundary of the sheet of paper, see Figure 3. They represent the singular points on the boundary of the surface. The model is constantly updated at each time step using adapted geometrical remeshing ensuring developability of the shape interleaved with a physically based elastic FEM simulation [ARC] guiding the main deformations on top of the coarse mesh.

Our tearing method can be seen as an additional feature of this existing pipe-line, which thus offers now a complete framework for interactive virtual paper manipulation. The deformation developed in [SRH*15] (both physical simulation step and geometric remeshing step) is used here to compute the interactive deformation of the surface without tearing consideration. Our method then computes the response of paper to extension forces by applying the following four steps, see also Figure 4:

(1) Selection of potential tearing points and fracture energy computation (Section 4)
A list of candidates to extend or start the tears propagation called

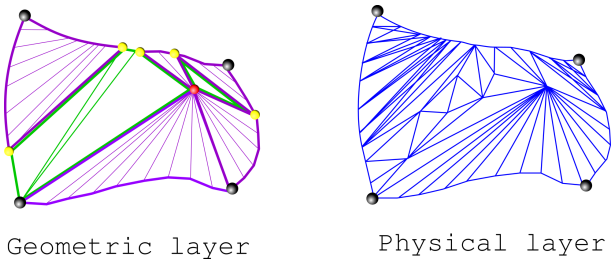


Figure 3: *Crumpled paper model from [SRH*15]: the geometric layer (left) is composed of curved regions (in purple) and flat regions (in green). Interior singular points (in red) represent damages in the fibrous structure. Singular points on the boundary, called junction points (in yellow), represent the junction between two region. The mesh used by the physical layer (right) is then obtained from the geometric layer to fit the folds*

potential tearing points is selected and associated to a controllable procedural formulation for the intrinsic fracture energy E_f.

The three following steps are then applied for each potential tearing point of the list.

(2) Hybrid model to compute tearing propagation (Section 5)
The local forces and the direction of the tear are efficiently computed based on the previously described assumptions. The release energy rate E_r is computed with respect to the applied physical forces, and the tear is then potentially propagated on the surface which is remeshed.

(3) Modeling the details along the tear trajectory (Section 6)
A high resolution path along the tear is computed using a texture based approach enabling to efficiently mimic detailed tearing appearance on top of a coarse geometrical mesh.

(4) A physical simulation step is finally applied on the mesh after each propagation of a tear, in order to dissipate the energy released by the propagation and let the lips of the tear move apart before treating the next potential tearing point. This is the same physical step than used in [SRH*15].

4. Potential tearing points and fracture energy

Finding the location of the starting position of a tear is a problem which has not been explicitly studied in Computer Graphics so far. Existing approaches consider either a predefined position which does not fit to an interactive model [KMB*09], or consider all vertices of the mesh as potential starting points for the tear leading to high computational cost [PNdJO14]. We propose instead to extract a limited subset of points where tearing may take place, and sort them with respect to a procedural criteria.

4.1. Selecting potential tearing points

Material sciences studies showed that inextensible thin sheets, including paper material, have a very low resistance to torque [Rom13]. At the opposite, they have a much higher resistance to in-plane stress. For instance, tearing a rectangular sheet of paper in-plane while keeping the sheet flat on a table is much more difficult compared to tearing a piece of paper by twisting one of its borders in two different directions as shown in Figure 5.

Taking into account the low resistance to torque and the fact that inextensible thin sheets concentrate stress in very localized singular regions [Fre98] we propose to consider the potential tearing points as being the vertices of the boundaries which may be able to concentrate the stress in our tearing scenarios. We identified three types of points
- The finger positions (red points in Figure 4 and Figure 5), where the hard positional constraints are enforced on the surface, therefore potentially accumulate large stress.
- The so-called *junction points* (yellow points in Figure 4) separating two curved regions (see Figure 3 and Figure 5), or a curved and a flat region on the boundary. These points are singular, and therefore concentrate the stress.
- The points exhibiting an inward angle on the boundary (sky blue points in Figure 4). Under deformation, these points separate different conical or planar surface parts and become therefore *junction points*. The tips of already existing tears fall in this category as well.

Note that following the underlying geometric model, which well approximates deformed paper sheets, all other points on the boundary are necessarily regular surface points belonging to smooth cones or planar regions, and therefore do not accumulate stress. We are therefore able to limit the number of potential tearing points which have to be checked at each time step to this small set of points, usually around 10 points in our examples.

4.2. Computing the fracture energy

Remember that the fracture energy E_f (1) represents the threshold above which the energy release rate E_r is sufficient to generate or propagate a tear. It is therefore related to the intrinsic resistance of the fiber network constituting the sheet of paper and needs to be estimated for each potential tearing point. We thus propose to model the resistance of the fibers by a 2D texture T_{fibers} over the surface. We use a Perlin noise as fibers' texture. Note that it would be possible to replace the Perlin noise by a more physically accurate representation of the distribution of the fibers as a future work.

In order to take into account the plastic damage occurring at the tip of a tear that creates microcracks propagating ahead of the tear tip, and thus weaken the structure in its neighborhood, We associate a damage parameter d_p to each potential tearing point $p = (u,v)$ (u and v being the coordinates of the point in the 2D pattern) which facilitates the propagation of an already existing tear. If p is a tip of a tear, we set $d_p = D$, where $D \in [0,1]$ is a constant value chosen by the user to define how easy the tear can be propagated. For all other points, we set $d_p = 1$. As shown in the accompanying video, the use of the damage parameter helps to ensure a continuous tear propagation in correspondence to the motion of the hands. The fracture energy of p is then defined as

$$E_f(p) = c\, T_{fiber}(u,v)\, d_p, \tag{3}$$

where $T_{fibers}(u,v) \in [0,1]$ is the value of the fibers texture at the position (u,v) and c is a material constant modeling the general resistance of the paper. Combined with the parameter D, c controls

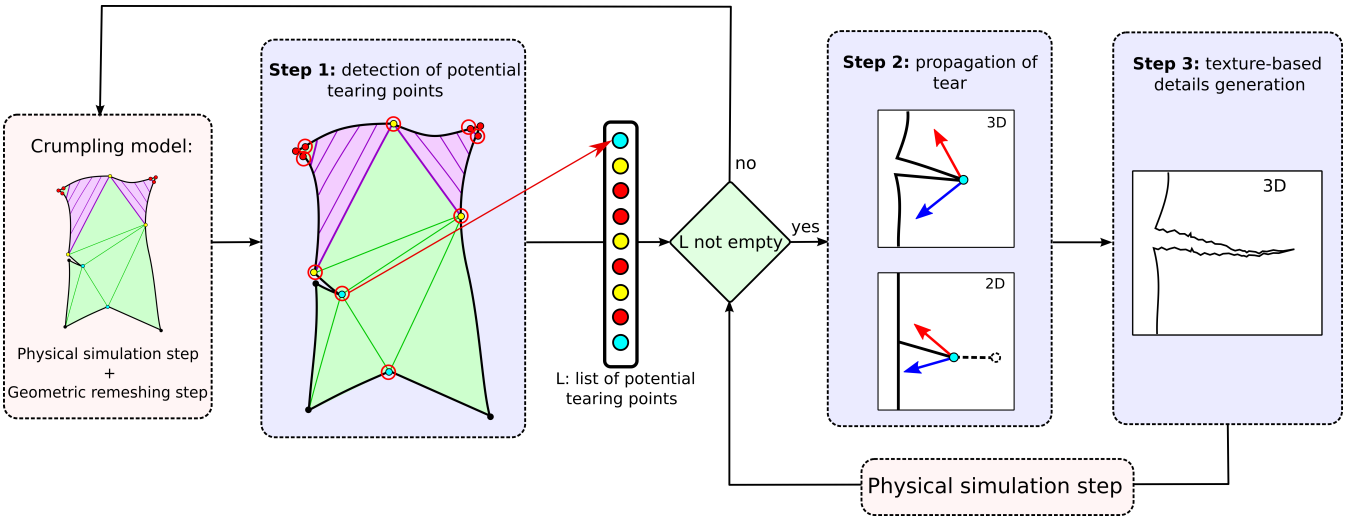


Figure 4: *Overview of our tearing method in three steps on top on existing bending and crumpling animation model.*

Figure 5: *Junction points (yellow point) in this real case scenario are specific points on the papers boundary from which a tearing path is supposed to start.*

how easily a tear will be created and how easily it will propagate. We use $c = 0.005$ in our examples.

5. Hybrid model to compute tearing propagation

Given a potential tearing point p and its fracture energy $E_f(p)$ (3) this section now describes how to compute the energy release rate $E_r(p)$ enabling to decide whether or not the tear path propagates at p and how to compute the direction of the path of the tear.

We call our approach *hybrid* (geometric and physical) because on one hand the underlying deformation model (crumpling model) [SRH*15] is hybrid and on the other hand we also interleave geometric and physical considerations for the tear path computations. The crumpling model has indeed two layers. The physical layer provides us with the forces derived from a standard elastic physical-based simulation step. The geometric layer consists in the developable surface of the sheet of paper defined by pieces of generalized cones through their set of rulings and planar surface pieces, see Figure 3. The geometric layer includes the tear path computations and is used to create an optimized mesh for the simulation step. This mesh is very coarse but has all the necessary degrees of freedom. Both the physical layer and the geometric layer of the deformation model are updated at each frame. See the overview Figure 4 to remember the big picture of the algorithm.

Our approach aims at integrating efficiently the geometrical and physical considerations noted by material sciences studies, in order to obtain the controllable, smooth paths specific to paper tearing. Despite the fact that the forces act on the tear in the 3D space, the tear trajectory seems to follow geometric rules on the 2D pattern. This is why, instead of choosing a 3D criterion as proposed in [OH99, PNdJO14] or a 2D energy based criterion as used by Gingold *et. al.* [GSH*04], we derive a criterion based both on the 3D forces and on 2D geometric rules. In addition, we can thus perform the tear path propagation and the mesh update in the 2D pattern, which simplifies considerably the computations in the geometric layer.

5.1. Tearing forces

Let us consider a potential tearing point p and the forces applied on it. These forces $\in \mathbb{R}^3$ are computed from the standard FEM used by the physical simulation step. Note that due to the very low bending forces of paper material, we consider only in-plane stretching forces during the tear propagation step which are the forces related to the deformation of the triangles of the mesh. If p, a vertex of the mesh, has N adjacent triangles, a set of N forces $(\mathbf{f}_k)_{k\in[0,N-1]}$ is applied on it, each one corresponding to an adjacent triangle. As stated previously in Section 3, we aim at efficiently approximate this local distribution of stretching forces by two equivalent (nearly) opposite forces called $\mathbf{F}_{left}$ and $\mathbf{F}_{right}$, that we consider as generated by two imaginary pulling points, so that we can apply the formula (2) derived by O'Keefe [OKe94]. Herein the direction of tearing is determined in the 2D pattern as the bisector of the pre-image of these two forces.

Our first goal is therefore to divide the forces $(\mathbf{f}_k)_{k\in[0,N-1]}$ into two clusters based on the similarity of their directions. For this, we look for a plane separating the forces defined by the normal vector $\mathbf{n}$ at position p and a "bisecting" vector $\mathbf{b}$. $\mathbf{F}_{left}$ and $\mathbf{F}_{right}$ are then defined by:

$$\mathbf{F}_{left} = \sum_{\mathbf{f}_k\cdot(\mathbf{b}\times\mathbf{n})>0} \mathbf{f}_k \text{ and } \mathbf{F}_{right} = \sum_{\mathbf{f}_k\cdot(\mathbf{b}\times\mathbf{n})<0} \mathbf{f}_k \,. \quad (4)$$

Our computation of $\mathbf{b}$ relies on an iterative clustering algorithm. First, we initiate $\mathbf{b}^0$ as the bisector of two tangent vectors on each side of p on the surface boundary, see Figure 6 (top). Note that these (generally not parallel) tangent vectors may correspond either to the initial surface boundary when a new tear is created, or the lips of an already existing tear. $\mathbf{F}^0_{left}$ and $\mathbf{F}^0_{right}$ are computed using Eq. (4). We then iterate over $i > 0$ by defining $\mathbf{b}^i$ as the bisector of $\mathbf{F}^{i-1}_{left}$ and $\mathbf{F}^{i-1}_{right}$ and pursue the computation of the forces $\mathbf{F}^i_{left}$ and $\mathbf{F}^i_{right}$ still using Eq. (4). Finally, we stop the algorithm when $\|\mathbf{b}^i - \mathbf{b}^{i-1}\| < \varepsilon$, which usually takes no more than 10 iterations.

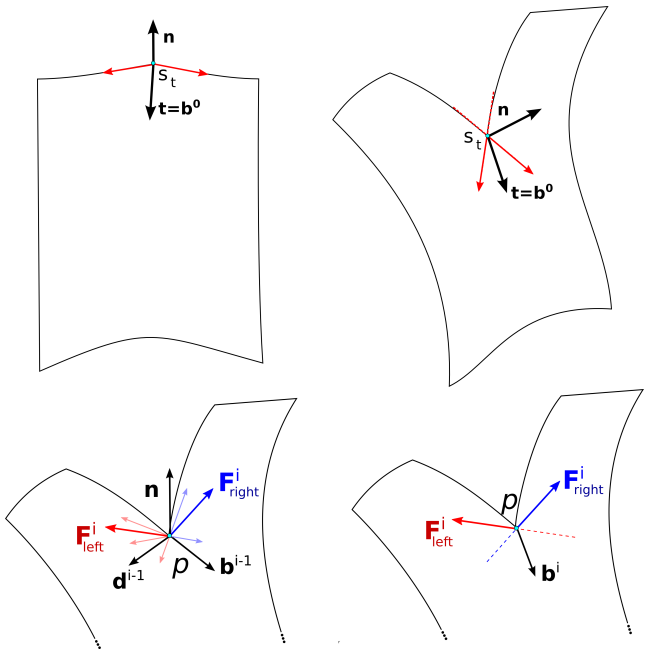


Figure 6: Top: *Initialization of tearing forces clustering algorithm. $\boldsymbol{b}^0$ is initiated as the bisector of the boundary tangents at p.* Bottom: *An iteration of the clustering algorithm. (left) $\boldsymbol{F}^{i+1}_{left}$ is computed as the sum of the forces (in red) of the left side of the plane defined by $\boldsymbol{b}^i$ and the normal to the surface at p and analogous for the forces on the right side (in blue). (right) $\boldsymbol{b}^{i+1}$ is then computed as the bisector of $\boldsymbol{F}^{i+1}_{left}$ and $\boldsymbol{F}^{i+1}_{right}$.*

5.2. Energy release rate and tearing direction

Once we have obtained the two clusters of 3D forces (represented by $\mathbf{F}_{left}$ and $\mathbf{F}_{right}$), we use them to compute the energy release rate and the direction of propagation as described before in Section 3.

The forces $(\mathbf{f}_k)$ composing the clusters are in-plane forces, so we can map each $\boldsymbol{f}_k$ into $\bar{\boldsymbol{f}}_k$ in the 2D pattern. We then compute $\bar{\mathbf{F}}_{left}$ and $\bar{\mathbf{F}}_{right}$ –the counterparts in the 2D pattern of $\mathbf{F}_{left}$ and $\mathbf{F}_{right}$– as the sum of mapped forces of each cluster:

$$\bar{\boldsymbol{F}}_{left} = \sum_{\mathbf{f}_k \cdot (\mathbf{b}\times\mathbf{n})>0} \bar{\boldsymbol{f}}_k \quad \text{and} \quad \bar{\boldsymbol{F}}_{right} = \sum_{\mathbf{f}_k \cdot (\mathbf{b}\times\mathbf{n})<0} \bar{\boldsymbol{f}}_k \,. \tag{5}$$

We consider $\bar{\mathbf{F}}_{left}$ and $\bar{\mathbf{F}}_{right}$ as the projections of forces on the pattern applied by the two imaginary pulling points.

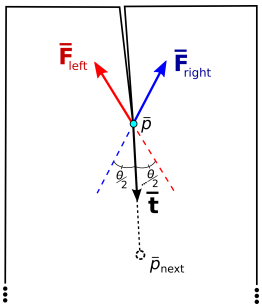


Figure 7: *The direction of propagation of a tear depends on the 2D angle θ between $\bar{\boldsymbol{F}}_{left}$ and $\bar{\boldsymbol{F}}_{right}$. If propagated, the next tip is found in the direction $\boldsymbol{t}$, the bisector of $\bar{\boldsymbol{F}}_{left}$ and $\bar{\boldsymbol{F}}_{right}$.*

The direction $\bar{\boldsymbol{t}}$ (shown in Figure 7) in which the tear should propagate to release the maximum of energy is then the bisector of $\bar{\mathbf{F}}_{left}$ and $\bar{\mathbf{F}}_{right}$ and the corresponding energy release rate (derived from (2)) is given by

$$E_r = (\|\mathbf{F}_{left}\| + \|\mathbf{F}_{right}\|) \cos\left(\frac{\theta}{2}\right)/e, \tag{6}$$

where θ is the angle $\in [0, 2\pi]$ between $\bar{\mathbf{F}}_{left}$ and $\bar{\mathbf{F}}_{right}$.

If the energy release rate E_r is greater than the fracture energy E_f at the point p (1), then the tear propagates and a new tip of the tear is created in the direction $\bar{\boldsymbol{t}}$. We adjust the speed of propagation according to the energy released by the propagation:

$$\bar{p}_{next} = \bar{p} + c\, E_r\, \bar{\boldsymbol{t}},$$

where c is a constant parameter enabling the user to tune the speed of propagation of the tear. Finally, the 3D position of the new tear tip is computed by mapping $\bar{p}_{next}$ back onto the 3D surface.

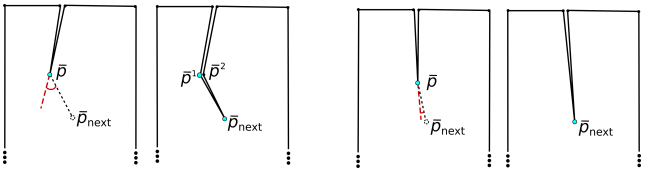


Figure 8: *Depending on the angle made by the tear at p, we either (left) split or (right) remove the vertex p*

5.3. Surface remeshing

Once the new tearing tip p_{next} has been found, the crumpling model, in particular the geometric layer, needs to be updated. As p is a point belonging to the boundary of the torn paper, p_{next} has to be integrated into the boundary of the surface. We distinguish two cases. When the tear direction in the 2D pattern changes at vertex $\bar{p}$ (i.e. the angle formed by the tear at p is greater than a threshold that we set at 0.1rad), we split p into two vertices, each belonging to one side of the tear lips, see Fig. 8(left). Otherwise, when the tear goes straight, we remove p from the boundary of the surface, see Fig. 8(right). This enable us to keep the mesh coarse in order to be as computational efficient as possible.

Finally, the coherence of the geometric layer has to be ensured: the generalized cones and flat regions crossed by the new tear between p and p_{next} have to be updated to integrate the tear. More technical details are given in Appendix A. Let us however notice, that one of two newly created vertices after splitting, p^1 or p^2, has necessarily an inward angle on the boundary, which means that it becomes a potential tearing point. The tear could thus even branch at this point in the next iterations of the tearing algorithm.

The physical layer is computed from the geometric layer and then automatically integrates the tear. The physical simulation is considered as static, so we put a zero velocity to every point created while remeshing.

6. Modeling the details along the tear trajectory

The previous section explained how we compute the next tip of a tear when it propagates during a small time step. This enables to obtain at a large scale a piecewise smoothly curved tearing path, such as the hyperbola-generated trajectories from [OKe94]. At a fine scale however, the border of a paper along a tear exhibits small stochastic details, as the ones that can observed on the real torn paper in Figure 9. In order to obtain realistic looking results with our method, these small stochastic details, need to be added. In this section, we explain how compute a detailed path between two successive tips and how to visualize it with a texture.

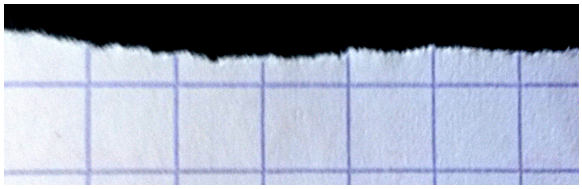


Figure 9: *Torn border of a real paper. The path of a paper tear presents small stochastic details that would be too expensive to precisely represent with the same mesh used for the modeling of deformation.*

6.1. Stochastic computation of the detailed path

To compute the fine scale geometry of the tear path between two successive tear tips i and $i+1$, we adapt the method from Lejemble *et al.* [LFD*15] by defining a 2D scalar mapping, which combines a stochastic distribution T_{fiber} representing the resistance of the fibers to break and a fracture-centered Gaussian stress field G_i:

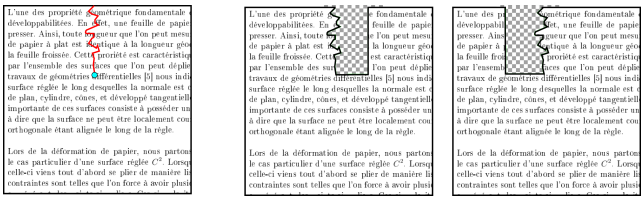
$$T(x,y) = (1-\omega)\,T_{fiber}(x,y) + \omega G_i(x,y). \qquad (7)$$

The size of the details are controlled by the standard deviation σ of the stress field G_i. T_{fiber} has already been introduced in formula (3) and causes the stochastic appearance of the resulting path. We model T by a 2D texture, which represents the breaking coefficient at each pixel (intuitively it can be seen as the probability of the path to go though a pixel). A low energy path between the positions of the two tips is then computed by iteratively choosing among the neighbor pixels that are closer to the next tip the one with the greatest breaking coefficient.

6.2. Representation of the detail path as textures

The detailed path between the two successive tips is described as a path inside a texture mapping the surface. Our objective is to efficiently render the fine detailed tear segments between successive tips following the global tear trajectory. A naive way to proceed would be to compute the position on the 2D pattern corresponding to each pixel of the path and then triangulate them and map them on the 3D surface. But this would require a very fine meshing near the tear path with an large number of triangles. Our idea is to keep the mesh coarse, and rather "cut" the path on the texture of the paper that we use when rendering our results.

Let us call the initial texture used to render the piece of paper T_{paper}. To effectively represent a detailed tearing path we modify this texture. When a new tear is created, we compute two textures from T_{paper}, T_{right} and T_{left}, that are respectively transparent on each side of the detailed path of tear (see Figure 10). The triangles that are adjacent to the tear are textured by either T_{right} or T_{left} according to which side of the tear they belong to.



T_{paper} T_{left} T_{right}

Figure 10: *The two textures T_{left} (middle) and T_{right} (right) used to render the fine details along the tear path as defined (in red) in the initial texture (left).*

When a tear is propagated, the textures T_{right} and T_{left} are updated in order to include the new path. When a tear is branching, two new textures are created for the new branch of the tear from the texture corresponding to the parent tear.

The detailed path between two successive tear tips is shared by two triangles. To have all the details of the path represented for both of them, we extend those triangles by a small textured rectangle as shown in Figure 11.

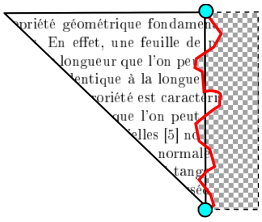


Figure 11: *A support rectangle is used to represent the details outside a triangle bordering a tear.*

7. Results

We present here some of our results using simple and easily reproducible scenarios. The animations corresponding to the examples presented in this section can be found in the accompanying video.

7.1. Validation of the starting points

To validate our method for computing the tearing starting point we performed an experimental evaluation. We compared the position of the starting point of the tear obtained by our virtual paper with the position for real paper in three cases: (a) the tear begins at a finger position, (b) the tear starts at a *junction point* and (c) at an inward angle of the boundary. These cases represent our three different types of potential tearing points on the boundary of the paper described in Section 4.1.

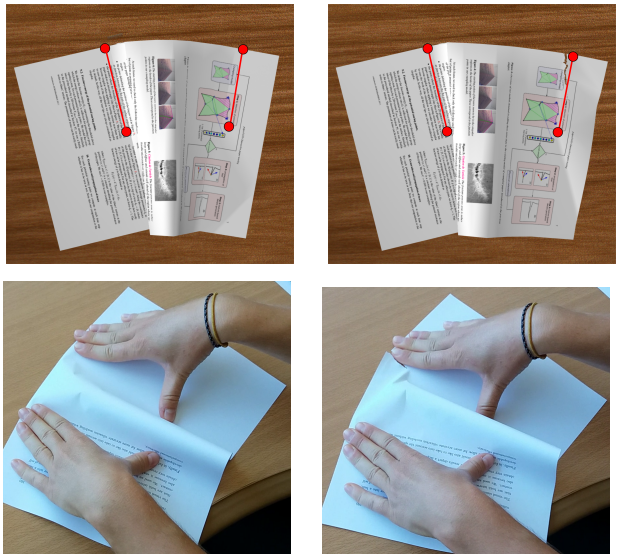


Figure 12: *A tear is created on virtual paper by flattening of the border of the bent surface. The deformation is generated by two hands represented each by two red points (modeling two fingers) linked by a red line. The tear appears at a junction point between the curved region and one of the flat regions on both, on our virtual sheet paper (top) as well as on the real one (bottom).*

Figure 12 shows a virtual piece of paper bent into a locally cylindrical shape by using two hands placed onto the sheet of paper. Each hand is represented on the figure by two red points linked by a red line. Then the two hands perform a rotation in order to flatten one border of the cylinder until creating a tear. The same experiment is realized with a real piece of paper. We can see that with both virtual and real paper the tear appears near one of the fingers on the border between the curved region and one of the parts maintained flat by a hand. Indeed, we assumed that fingers, modeled as hard positional constraint, can potentially accumulate large stress, leading easily to tears.

Figure 13 shows an example of a tear created by twisting the border of a piece of paper. Two corners of the paper are hold and rotated in opposite directions until two curved regions are obtained, curved in opposite directions. In both cases, for the virtual and the real paper, we make the same observations. First a singularity appears at the junction between the two curved regions. In our virtual model, this is identified as a *junction point* (represented in the virtual model as a yellow point in Figure 5). Then a tear starts at this point and continues to propagate inside the paper.

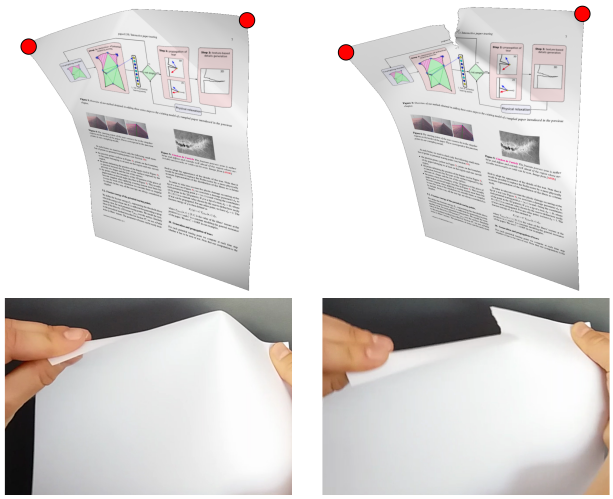


Figure 13: *A tear is created by bending two corners of the sheet of paper in two different directions,i.e. by twisting the border. On both our virtual paper (top) and real paper (bottom), the tear appears at the junction between the two cones created by each bending.*

In the third scenario, cut out a triangular piece of paper and therefore create an inward angle at the boundary, see Figure 14 and Figure 15. We then tear the paper at this concave border by placing the fingers at the top and by pulling the extremities in opposite directions. In this situation the point at the inward angle easily becomes a junction point and so easily becomes the starting point of a tear. We reproduce this case in Figure 14 and Figure 15.

It is possible the check the validity of our potential tearing points be reproducing the scenarios presented above.

These results suggest that the three types of particular points on the geometric model (finger positions, junction points and concave points), which we have selected with physical and geometric arguments do work well as potential starting tearing points. We note, that real paper can also be torn starting from the inside of the surface when forcing an object though it. However, this kind of scenario is out of scope of the present paper, as we aimed at reproducing controllable tears when manipulating a piece of paper with our fingers. The tears caused by this particular case are difficult to create solely by hand and appear abruptly without being controllable.

7.2. Validation of the tear trajectory

As explained in Section 3, and in more details in [Rom13] and [OKe94], when the propagation of the tear is only caused by two points being pulled apart, the path drawn by the tear on the 2D pattern follows a very particular geometric curve. Locally, at each tear tip p, the path follows the tangent of a hyperbolic curve whose focal points are the fingers positions (see Figure 2) which is the bisector of the two line segments linking the two pulling points to p. In the next example, we have reproduced the experiment reported in [OKe94] in order to validate our more general model, which has been derived from these results.

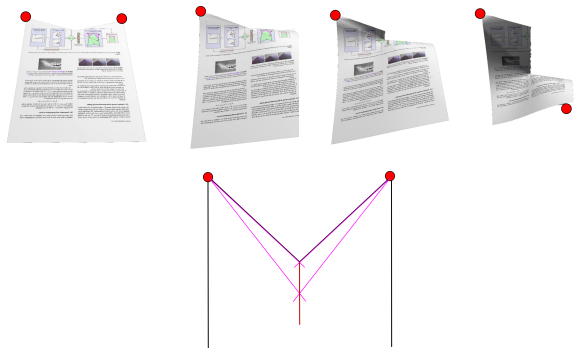


Figure 14: *Two pulling points at equal distance of the starting point of the tear create a straight tear. (top) Evolution of the 3D surface as the tear propagates. (bottom) 2D pattern, the tear path is the mediator of the two pulling points.*

Figure 14 shows the paper being torn by pulling two points (red dots) located at equal distances of the starting point of the tear. To prepare the sheet of paper, we cut out a triangular piece.This enable us to predict exactly where the tear will start and to place the imaginary fingers at equal distance to the tear's starting point. Following the theory [OKe94, Rom13], the energy release rate is maximized in the direction of the bisector of the 2 line segments. As we artificially created a symmetric situation, it is expected that the tear trajectory goes straight. Indeed, the tear shown in Figure 14 follows the mediator of the line segments between the two pulling points describing therefore the expected straight tear trajectory.

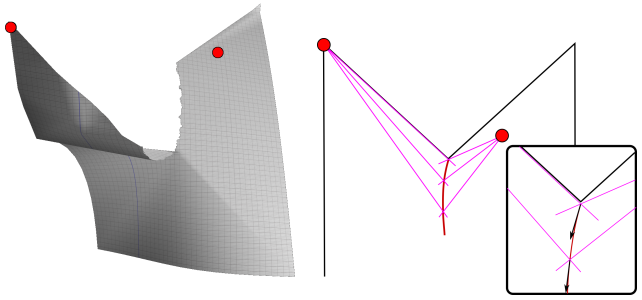


Figure 15: *Two pulling points at different distances of the starting point of the tear create a curved tear. (left) The 3D surface of the torn paper. (right) 2D pattern, at each point, the tangent to the path approximately cuts the angle between the directions toward each pulling points into two equal part.*

7.3. Other examples

In the next example, our intention was to reproduce an asymmetric case leading to a curved path. Figure 15 shows the tear obtained by pulling two points located at different distances from the starting points. The mapping of the tear correspond closely to the expected hyperbolic trajectory and can be observed in the video.

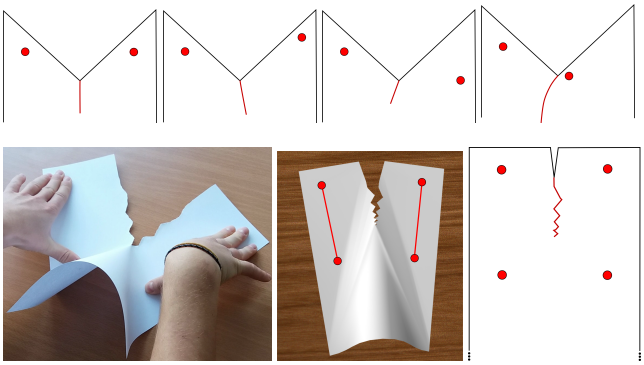


Figure 16: *Top: By choosing the position of two pulling points, we can control the direction of the tear. Bottom: We use the thumb and forefinger, represented as red dots on the virtual paper and the corresponding 2D pattern (right) of each hand to tear the paper. By rotating them successively we obtain the same zigzagging curve for the real paper (left) as well as on our virtual paper (middle).*

User control was one of our main goals when developing our algorithm. It turns out that the user can indeed intuitively control the path of the tear by choosing the positions for pulling fingers as shown in Figure 16(top). This enables us to create complex tearing paths, such as the zigzag shown in Figure 16(bottom). The zigzag was obtained by putting the thumb and the forefinger of each hand onto the paper and rotating then successively one hand after the other. We can also obtain a tear with several branches as shown in Figure 17, when using a more general scenario where fingers change positions through the deformation. We thus demonstrate, that a fine and direct user control over the tearing path is possible and that our model behaves as expected in these scenarios.

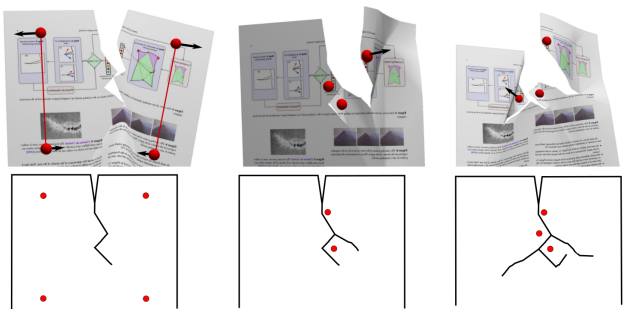


Figure 17: *A branching tear is obtained by tearing the paper successively into different directions using the fingers represented by red dots.*

Figure 18 shows an example of a more complex deformation involving crumpling and tearing. A sheet of paper is bent in a cylindrical shape by two hands (one of each side of the curved part) (left). A border is then pinched by one of the hands (middle-left). The hands move apart of each other by a rotational motion, creating a singular point on the border and then a tear starting from this point (middle-right, right).

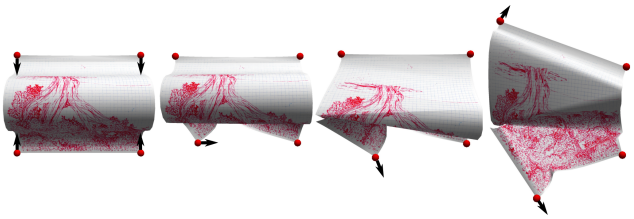


Figure 18: *A more complex example exhibiting crumpling and tearing. Red dots are finger points and black arrows indicate their motion.*

7.4. Details

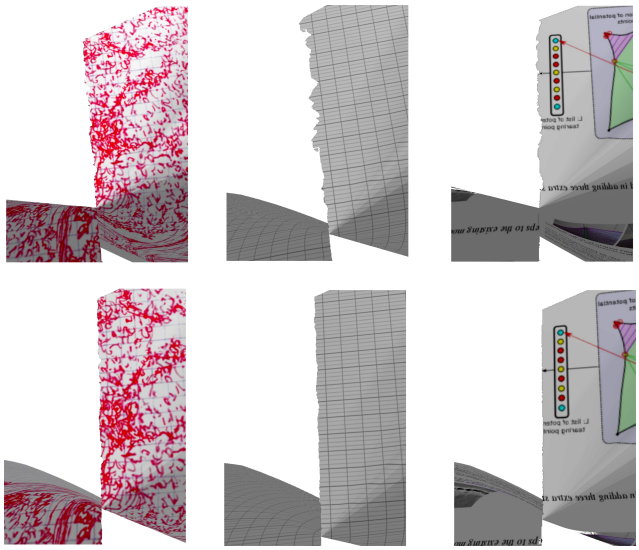


Figure 19: *Details obtained using different parameters. The standard deviation σ of the Gaussian field in (7) between two successive tips controls the size of the details. (first row) $\sigma = 0.001$. (second row) $\sigma = 0.0001$. The resolution of the texture influences the resolution of the details. Size of the texture: (left column) 4528×3200, (middle column) 2048×1476, (right column) 1169×826.*

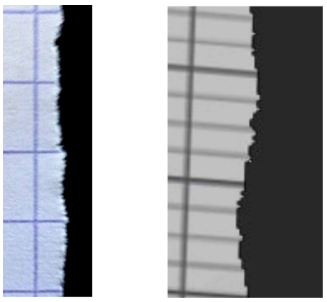


Figure 20: *Close view on a torn border of real paper (left) and one of our virtual paper (right). Each image represents around 2cm of the torn border.*

Figure 19 gives a closer view on stochastic details and shows that we can obtain small details without requiring a dense mesh. Varying the size of the resolution and the standard deviation of the Gaussian field G_i in (7) enables to modify the appearance of the tear.

For instance, with a texture resolution of 2048×1476 corresponding to a dinA4 sheet of paper (21×29.7 cm), and a standard deviation $\sigma = 0.1mm$ (see Figure 19 (bottom-middle), we can obtain details that are of the same scale than the real paper shown in Figure 9. Figure 20 shows a close view on the details of real and virtual paper, along 2 cm of torn border.

7.5. Comparison with the approach of Pfaff *et al.* [PNdJO14]

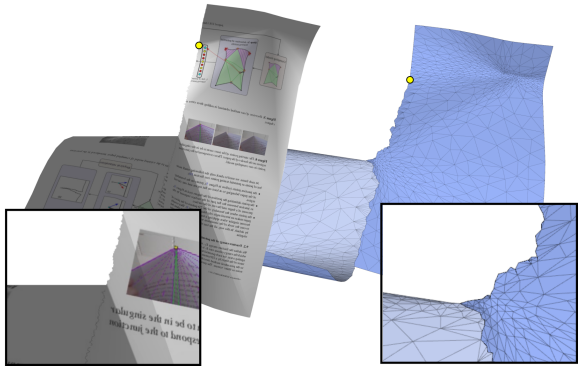


Figure 21: *Comparison between a tear obtained with our method (left) and one obtained with the method presented in [PNdJO14] (right). The tear is created from a notch with two fingers at equal distance of the end of the notch (yellow dot).*

Although the method presented by Pfaff *et. al.* [PNdJO14] shows convincing results for many cases of tearing thin shells, it is not well-adapted for the specific case of paper tearing and in particular for interactive manipulations. It not only requires a dense mesh at the tip of the tear –making it computationally rather expensive– but also it cannot reproduce the predictable, smooth-looking aspect of tears in paper. Figure 21 shows an example of this effect: a tear is created by pulling apart two points placed at equal distance of a notch. The tear obtained with the method from [PNdJO14] follows a oscillating slightly curved path that is not controlled by the position of the fingers. On the opposite, our method enables to obtain the expected straight smooth tear trajectory, to which details of various aspects may then be added and parameterized.

7.6. Performance

The Table 1 shows the computation times obtained on a laptop with 8 cores (2.70GHz). The average and maximal times per frame are computed for the animations presented in our results. We obtain an animation rate between 10 and 3 frames per second, which enables interactive applications. The bottleneck of the computational time is still the physical simulation used to compute the underlying elastic deformation [ARC]. Notably, we use a small time step of simulation (in the order of 1/10th of the visual time frame) in

order to simulate rigid behaviors. Still, we managed with our tearing method to keep the mesh, which is also used in the simulation, very coarse (in the order of a hundred triangles), so that the computational cost of the physical simulation stays reasonable for interactivity.

	#Δ	Time spent in each step (*in ms per frame*)					
		Physical simulation		Geometrical remeshing		Tearing computation	
		avg	*max*	*avg*	*max*	*avg*	*max*
Fig. 12	99	248	301	7	11	2	12
Fig. 13	70	119	206	4	14	2	15
Fig. 14	68	170	244	4	5	3	7
Fig. 16	102	254	303	8	17	6	17

Table 1: ***Average and maximum computation times*** *for the respective steps: physical simulation [ARC] (including both the physical step use in [SRH*15] and the one after each propagation of a tear), geometric remeshing of the crumpling method [SRH*15] and for the tearing step described in this article (steps 1, 2, and 3 from Fig 4). The second column indicates the average number of triangles through all the frames.*

8. Discussion and conclusion

We presented a method to interactively model a piece of paper being torn. By separating the generation of details and the global trajectory of the tear, we are able to obtain tears whose general path follows a smooth and predictable tear, and which at the same time presents small, refined details. The geometrical-based tearing paths are consistent with respect to behaviors described in the physics and fracture mechanics literature and validated with experiments on real paper.

In this work, we focused on efficient computation and intuitively controllable scenarios on inextensible thin sheet material, our method has then several limitations in the range of applicability that may be addressed in future works.

We based our algorithm on assumptions that are very specific to paper-like material, notably the ones from O'Keefe's work used for the propagation of the tear. Such behavior may not be directly applicable to elastic material such as cloth for instance. Also the described approach has been designed to handle tears caused by a torque imposed on a boundary of the surface, which are the tears that occur most often while tearing paper in real life. This assumption prohibits our model to start a tear inside the surface, such as caused by a projectile. Initializing a tear by using two perfectly colinear forces is also out of the scope of the possible scenarios. Note that in these cases, the sheet of paper would get brutally torn and the tear would hardly be controllable. Still, once initiated, our general idea for computing the tearing path in clustering forces and computing direction within the 2D pattern may be used to pursue the tearing process.

As visible in the videos, some sudden geometrical changes may occur during animation. They are related to the remeshing of a very coarse mesh structure and could be limited by using dedicated smoothing such as, for instance, the post-remeshing projection proposed by Narain *et al.* [NSO12].

In the near future, we plan to improve the details appearance, e.g. by generating damages in the fibers' texture around singular regions. It would thus be possible to control where the fibrous structure of the paper is weaker, in order to simulate the effect of broken fibers. In order to model plastic deformation effects such as an existing fold line, we also plan either to modify the fiber texture along the fold to influence the detailed path, or to influence the general path by introducing a bias in the choice of the direction of propagation described in Section 5.2. Another idea would be to use several fiber textures to represent multiple layers of fibers. In this way, the same tear would be associated to several detailed paths and then be represented by layered semi-transparent textures in order to improve realism.

Acknowledgements: This research was partially funded by the ERC advanced grant no. 291184 EXPRESSIVE.
We thank Estelle Charleroy for the help on the rendering and video montage.

References

[AL85] AMESTOY M., LEBLOND J.-B.: Damage, fatigue, fracture. - on the criterion giving the direction of propagation of cracks in the griffith theory. *C. R. Acad. Sciences, Série II 301* (1985), 969–972. 3

[AN06] ALAVA M., NISKANEN K.: The physics of paper. *Reports on Progress in Physics 69*, 3 (2006), 669.

[ARC] ARCSIM: http://graphics.berkeley.edu/resources/arcsim/. 3, 10, 11

[ARR05] AUDOLY B., REIS P. M., ROMAN B.: Cracks in thin sheets: When geometry rules the fracture path. *Physiscal Review Letters 95* (Jul 2005), 025502. 2

[BDW13] BUSARYEV O., DEY T. K., WANG H.: Adaptive fracture simulation of multi-layered thin plates. *ACM Transactions on Graphics 32*, 4 (July 2013), 52:1–52:6. 2

[BFM08] BOURDIN B., FRANCFORT G. A., MARIGO J.-J.: The variational approach to fracture. *Journal of elasticity 91*, 1-3 (2008), 5–148. 3

[BHTF07] BAO Z., HONG J.-M., TERAN J., FEDKIW R.: Fracturing rigid materials. *IEEE Transactions on Visualization and Computer Graphics 13*, 2 (Mar. 2007), 370–378. 2

[BW07] BO P., WANG W.: Geodesic-controlled developable surfaces for modeling paper bending. *Computer Graphics Forum 26*, 3 (2007). 2

[CFM09] CHAMBOLLE A., FRANCFORT G., MARIGO J.-J.: When and how do cracks propagate? *Journal of the Mechanics and Physics of Solids 57*, 9 (2009), 1614 – 1622. 3

[CYFW14] CHEN Z., YAO M., FENG R., WANG H.: Physics-inspired adaptive fracture refinement. *ACM Transactions on Graphics 33*, 4 (July 2014), 113:1–113:7. 2

[Fre98] FREUND L. B.: *Dynamic fracture mechanics*. Cambridge university press, 1998. 2, 4

[GMD12] GLONDU L., MARCHAL M., DUMONT G.: Real-Time Simulation of Brittle Fracture using Modal Analysis. *IEEE Transactions on Visualization and Computer Graphics 19*, 2 (Aug. 2012), 201–209. 2

[Gri21] GRIFFITH A. A.: The phenomena of rupture and flow in solids. *Philosophical transactions of the royal society of london. Series A, containing papers of a mathematical or physical character 221* (1921), 163–198. 3

[GSH*04] GINGOLD Y., SECORD A., HAN J. Y., GRINSPUN E., ZORIN D.: A discrete model for inelastic deformation of thin shells. *Technical Report* (2004). 2, 5

[HJST13] HEGEMANN J., JIANG C., SCHROEDER C., TERAN J. M.: A level set method for ductile fracture. In *Proceedings of the 12th ACM SIGGRAPH/Eurographics Symposium on Computer Animation* (New York, NY, USA, 2013), SCA '13, ACM, pp. 193–201. 2

[HY14] HWANG H.-D., YOON S.-H.: Constructing developable surfaces by wrapping cones and cylinders. *Solid and Physical Modeling 58* (2014). 2

[KMB*09] KAUFMANN P., MARTIN S., BOTSCH M., GRINSPUN E., GROSS M.: Enrichment textures for detailed cutting of shells. *ACM Transactions on Graphics 28*, 3 (July 2009), 50:1–50:10. 2, 4

[KZC09] KANG Y.-M., ZHANG H.-G., CHO H.-G.: Plausible virtual paper for real-time application. *CASA (Short Paper)* (2009). 2

[LFD*15] LEJEMBLE T., FONDEVILLA A., DURIN N., BLANC-BEYNE T., SCHRECK C., MANTEAUX P.-L., KRY P. G., CANI M.-P.: Interactive procedural simulation of paper tearing with sound. In *Motion In Games (MIG)* (Paris, France, Nov. 2015). 2, 7

[LG03] LEBLOND J.-B., GERMAIN P.: *Mécanique de la rupture fragile et ductile*. Hermès science publications, 2003. 2

[MBCN09] METAAPHANON N., BANDO Y., CHEN B.-Y., NISHITA T.: Simulation of tearing cloth with frayed edges. In *Computer Graphics Forum* (2009), vol. 28, Wiley Online Library, pp. 1837–1844. 2

[MCK13] MÜLLER M., CHENTANEZ N., KIM T.-Y.: Real time dynamic fracture with volumetric approximate convex decompositions. *ACM Transactions on Graphics 32*, 4 (July 2013), 115:1–115:10. 2

[NF99] NEFF M., FIUME E.: A visual model for blast waves and fracture. In *Proceedings of the 1999 Conference on Graphics Interface '99* (San Francisco, CA, USA, 1999), Morgan Kaufmann Publishers Inc., pp. 193–202. 2

[NSO12] NARAIN R., SAMII A., O'BRIEN J. F.: Adaptive anisotropic remeshing for cloth simulation. *ACM Transactions on Graphics (SIGGRAPH Asia Proc.) 31*, 6 (2012). 2, 11

[OBH02] O'BRIEN J. F., BARGTEIL A. W., HODGINS J. K.: Graphical modeling and animation of ductile fracture. In *Proceedings of ACM SIGGRAPH* (Aug. 2002), ACM Press, pp. 291–294. 2

[OH99] O'BRIEN J. F., HODGINS J. K.: Graphical modeling and animation of brittle fracture. In *Proceedings of ACM SIGGRAPH* (Aug. 1999), ACM Press/Addison-Wesley Publishing Co., pp. 137–146. 2, 5

[OKe94] OKEEFE R.: Modeling the tearing of paper. *American Journal of Physics 62*, 4 (1994), 299–305. 2, 3, 5, 7, 8, 9

[PNdJO14] PFAFF T., NARAIN R., DE JOYA J. M., O'BRIEN J. F.: Adaptive tearing and cracking of thin sheets. *ACM Transactions on Graphics 33*, 4 (July 2014), xx:1–9. 2, 4, 5, 10

[Rom13] ROMAN B.: Fracture path in brittle thin sheets: a unifying review on tearing. *International Journal of Fracture 182*, 2 (2013), 209–237. 2, 3, 4, 8, 9

[SP74] SETH R., PAGE D.: Fracture resistance of paper. *Journal of Materials Science 9*, 11 (1974), 1745–1753. 3

[SRH*15] SCHRECK C., ROHMER D., HAHMANN S., CANI M.-P., JIN S., WANG C. C.-L., BLOCH J.-F.: Non-smooth developable geometry for interactive paper crumpling. *ACM Transactions on Graphics 35*, 1 (2015). 2, 3, 4, 5, 11, 12

[TF88] TERZOPOULOS D., FLEISCHER K.: Modeling inelastic deformation: Viscolelasticity, plasticity, fracture. *SIGGRAPH Comput. Graph. 22*, 4 (June 1988), 269–278. 2

[ZBG15] ZHU Y., BRIDSON R., GREIF C.: Simulating rigid body fracture with surface meshes. *ACM Transactions on Graphics (SIGGRAPH Proc.) 34*, 4 (2015). 2

Appendix A: Geometry update after tear propagation

The geometric structure used by the crumpling model should be preserved. The geometry update can therefore be divided into three cases. For more details the reader is referred to [SRH*15].

Case 1: p_{next} is inside a generalized cone (i.e. $\bar{p}_{next}$ is inside the 2D pattern of the cone) whose apex is p, in which case we modify the cone such that its apex becomes p_{next} (see Figure 22).

Case 2: p_{next} is inside a curved region and p is not the apex of a generalized cone. We apply a remeshing scheme similar to the one used when creating a singular point inside a curved region (see Figure 23).

Case 3: p_{next} is inside a flat region. We just update the coarse triangulation representing the flat region in the geometric structure (see Figure 24).

Note also that if p is the apex of a generalized cone, we updated the regions such that p_{next} becomes the apex of this cone.

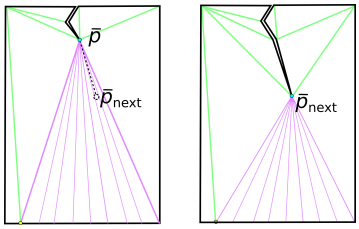


Figure 22: *Case 1 of the remeshing scheme after the propagation of a tear: p is the apex of a cone and p_{next} is inside this cone. After remeshing, p_{nxt} is the apex.*

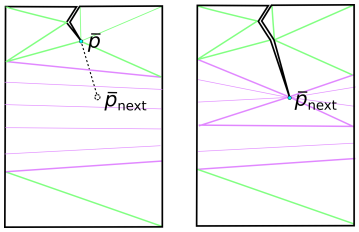


Figure 23: *Case 2: p_{next} is inside a curved region.*

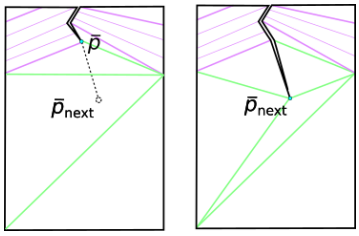


Figure 24: *Case 3: p_{next} is inside a flat region.*